WEST Search History

Hide Items | Restore | Clear | Cancel

DATE: Tuesday, May 24, 2005

Hide?	Set Name	Query	Hit Count
		DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=ADJ	
☐	L6	L5 and garbage collection	61
☐	L5	719/316,330.ccls.	726
☐	L4	L3 and garbage collection	337
☐	L3	707/206.ccls.	529
☐	L2	L1 and garbage collection	165
☐	L1	717/127,114-118,148,153,162-167,149,136.ccls.	2482

END OF SEARCH HISTORY

Terms used **java garbage collection** Found **66** of **154,226**

Sort results by: relevance
Display results: expanded form

Save results to a Binder
Search Tips
☐ Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 20 of 66 Result page: **1** 2 3 4 next

Relevance scale

1 Controlling garbage collection and heap growth to reduce the execution time of Java applications
Tim Brecht, Eshrat Arjomandi, Chang Li, Hang Pham
October 2001 **ACM SIGPLAN Notices , Proceedings of the 16th ACM SIGPLAN conference on Object oriented programming, systems, languages, and applications**, Volume 36 Issue 11
Full text available: pdf(723.07 KB) Additional Information: full citation, abstract, references, citings, index terms

In systems that support garbage collection, a tension exists between collecting garbage too frequently and not collecting garbage frequently enough. Garbage collection that occurs too frequently may introduce unnecessary overheads at the rist of not collecting much garbage during each cycle. On the other hand, collecting garbage too infrequently can result in applications that execute with a large amount of virtual memory (i.e., with a large footprint) and suffer from increased execution times d ...

2 Myths and realities: the performance impact of garbage collection
Stephen M. Blackburn, Perry Cheng, Kathryn S. McKinley
June 2004 **ACM SIGMETRICS Performance Evaluation Review , Proceedings of the joint international conference on Measurement and modeling of computer systems**, Volume 32 Issue 1
Full text available: pdf(305.06 KB) Additional Information: full citation, abstract, references, citings, index terms

This paper explores and quantifies garbage collection behavior for three whole heap collectors and generational counterparts: *copying semi-space, mark-sweep,* and *reference counting*, the canonical algorithms from which essentially all other collection algorithms are derived. Efficient implementations in MMTk, a Java memory management toolkit, in IBM's Jikes RVM share all common mechanisms to provide a clean experimental platform. Instrumentation separates collector and program behav ...

Keywords: generational, java, mark-sweep, reference counting, semi-space

3 Estimating the impact of heap liveness information on space consumption in Java
Ran Shaham, Elliot K. Kolodner, Mooly Sagiv
June 2002 **ACM SIGPLAN Notices , Proceedings of the 3rd international symposium on Memory management**, Volume 38 Issue 2 supplement

Full text available: pdf(303.65 KB) Additional Information: full citation, abstract, references, index terms

We study the potential impact of different kinds of liveness information on the space consumption of a program in a garbage collected environment, specifically for Java. The idea is to measure the time difference between the actual time an object is collected by the garbage collector (GC) and the potential earliest time an object could be collected assuming liveness information were available. We focus on the following kinds of liveness information: (i) stack-reference liveness (local reference ...

Keywords: Java, compilers, garbage collection, liveness analysis, memory management, program analysis

4 A generational mostly-concurrent garbage collector
Tony Printezis, David Detlefs
October 2000 **ACM SIGPLAN Notices , Proceedings of the 2nd international symposium on Memory management**, Volume 36 Issue 1
Full text available: pdf(1.67 MB) Additional Information: full citation, abstract, citings, index terms

This paper reports our experiences with a mostly-concurrent incremental garbage collector, implemented in the context of a high performance virtual machine for the Java™ programming language. The garbage collector is based on the "mostly parallel" collection algorithm of Boehm *et al.* and can be used as the old generation of a generational memory system. It overloads efficient write-barrier code already generated to support generational garbage collection to also ident ...

5 Heap compression for memory-constrained Java environments
G. Chen, M. Kandemir, N. Vijaykrishnan, M. J. Irwin, B. Mathiske, M. Wolczko
October 2003 **ACM SIGPLAN Notices , Proceedings of the 18th annual ACM SIGPLAN conference on Object-oriented programing, systems, languages, and applications**, Volume 38 Issue 11
Full text available: pdf(2.14 MB) Additional Information: full citation, abstract, references, citings, index terms

Java is becoming the main software platform for consumer and embedded devices such as mobile phones, PDAs, TV set-top boxes, and in-vehicle systems. Since many of these systems are memory constrained, it is extremely important to keep the memory footprint of Java applications under control.The goal of this work is to enable the execution of Java applications using a smaller heap footprint than that possible using current embedded JVMs. We propose a set of memory management strategies to reduce h ...

Keywords: Java virtual machine, garbage collection, heap, memory compression

6 Memory system behavior of Java programs: methodology and analysis
Jin-Soo Kim, Yarsun Hsu
June 2000 **ACM SIGMETRICS Performance Evaluation Review , Proceedings of the 2000 ACM SIGMETRICS international conference on Measurement and modeling of computer systems**, Volume 28 Issue 1
Full text available: pdf(1.08 MB) Additional Information: full citation, abstract, references, citings, index terms

This paper studies the memory system behavior of Java programs by analyzing memory reference traces of several SPECjvm98 applications running with a Just-In-Time (JIT) compiler. Trace information is collected by an exception-based tracing tool called JTRACE, without any instrumentation to the Java programs or the JIT compiler.First, we find that the overall cache miss ratio is increased due to garbage collection, which suffers from higher cache misses compared to the application. ...

7 Automated discovery of scoped memory regions for real-time Java
Morgan Deters, Ron K. Cytron
June 2002 **ACM SIGPLAN Notices , Proceedings of the 3rd international symposium on Memory management**, Volume 38 Issue 2 supplement
Full text available: pdf(227.49 KB) Additional Information: full citation, abstract, references, citings, index terms

Advances in operating systems and languages have brought the ideal of reasonably-bounded execution time closer to developers who need such assurances for real-time and embedded systems applications. Recently, extensions to the Java libraries and virtual machine have been proposed in an emerging standard, which provides for specification of release times, execution costs, and deadlines for a restricted class of threads. To use such features, the code executing in the thread must never reference s ...

Keywords: garbage collection, memory management, real-time Java, regions, trace-based analysis

8 Beltway: getting around garbage collection gridlock
Stephen M Blackburn, Richard Jones, Kathryn S McKinley, J Eliot B Moss
May 2002 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2002 Conference on Programming language design and implementation**, Volume 37 Issue 5
Full text available: pdf(184.50 KB) Additional Information: full citation, abstract, references, citings, index terms

We present the design and implementation of a new garbage collection framework that significantly generalizes existing copying collectors. The *Beltway* framework exploits and separates object age and incrementality. It groups objects in one or more increments on queues called *belts*, collects belts independently, and collects increments on a belt in first-in-first-out order. We show that Beltway configurations, selected by command line options, act and perform the same as semi-space, ...

Keywords: Java, beltway, copying collection, generational collection

9 Tuning garbage collection for reducing memory system energy in an embedded java environment
G. Chen, R. Shetty, M. Kandemir, N. Vijaykrishnan, M. J. Irwin, M. Wolczko
November 2002 **ACM Transactions on Embedded Computing Systems (TECS)**, Volume 1 Issue 1
Full text available: pdf(740.23 KB) Additional Information: full citation, abstract, references, citings, index terms

Java has been widely adopted as one of the software platforms for the seamless integration of diverse computing devices. Over the last year, there has been great momentum in adopting Java technology in devices such as cellphones, PDAs, and pagers where optimizing energy consumption is critical. Since, traditionally, the Java virtual machine (JVM), the cornerstone of Java technology, is tuned for performance, taking into account energy consumption requires reevaluation, and possibly redesign of t ...

Keywords: Garbage collector, Java Virtual Machine (JVM), K Virtual Machine (KVM), low power computing

10 Mostly concurrent garbage collection revisited
Katherine Barabash, Yoav Ossia, Erez Petrank

October 2003 **ACM SIGPLAN Notices , Proceedings of the 18th annual ACM SIGPLAN conference on Object-oriented programing, systems, languages, and applications**, Volume 38 Issue 11

Full text available: pdf(279.42 KB) Additional Information: full citation, abstract, references, citings, index terms

The *mostly concurrent garbage collection* was presented in the seminal paper of Boehm et al. With the deployment of Java as a portable, secure and concurrent programming language, the mostly concurrent garbage collector turned out to be an excellent solution for Java's garbage collection task. The use of this collector is reported for several modern production Java Virtual Machines and it has been investigated further in academia.In this paper, we present a modification of the mostly concu ...

Keywords: JVM, Java, concurrent garbage collection, garbage collection, incremental garbage collection

11 Mark-copy: fast copying GC with less space overhead
Narendran Sachindran, J. Eliot, B. Moss
October 2003 **ACM SIGPLAN Notices , Proceedings of the 18th annual ACM SIGPLAN conference on Object-oriented programing, systems, languages, and applications**, Volume 38 Issue 11

Full text available: pdf(297.93 KB) Additional Information: full citation, abstract, references, citings, index terms

Copying garbage collectors have a number of advantages over non-copying collectors, including cheap allocation and avoiding fragmentation. However, in order to provide completeness (the guarantee to reclaim each garbage object eventually), standard copying collectors require space equal to twice the size of the maximum live data for a program. We present a *mark-copy* collection algorithm (MC) that extends generational copying collection and significantly reduces the heap space required to ...

Keywords: Java, copying collector, generational collector, mark-copy, mark-sweep

12 Older-first garbage collection in practice: evaluation in a Java Virtual Machine
Darko Stefanović, Matthew Hertz, Stephen M. Blackburn, Kathryn S. McKinley, J. Eliot B. Moss
June 2002 **ACM SIGPLAN Notices , Proceedings of the workshop on Memory system performance**, Volume 38 Issue 2 supplement

Full text available: pdf(1.15 MB) Additional Information: full citation, abstract, references, citings

Until recently, the best performing copying garbage collectors used a generational policy which repeatedly collects the very youngest objects, copies any survivors to an older space, and then infrequently collects the older space. A previous study that used garbage-collection simulation pointed to potential improvements by using an *Older-First* copying garbage collection algorithm. The Older-First algorithm sweeps a fixed-sized window through the heap from older to younger objects, and avo ...

13 Controlling fragmentation and space consumption in the metronome, a real-time garbage collector for Java
David F. Bacon, Perry Cheng, V. T. Rajan
June 2003 **ACM SIGPLAN Notices , Proceedings of the 2003 ACM SIGPLAN conference on Language, compiler, and tool for embedded systems**, Volume 38 Issue 7

Full text available: pdf(354.15 KB) Additional Information: full citation, abstract, references, citings, index terms

Now that the use of garbage collection in languages like Java is becoming widely accepted due to the safety and software engineering benefits it provides, there is significant interest in applying garbage collection to hard real-time systems. Past approaches have generally suffered from one of two major flaws: either they were not provably real-time, or they imposed large space overheads to meet the real-time bounds.Our previous work [3] presented the Metronome, a mostly non-copying real-time co ...

Keywords: compaction, cost model, fragmentation, space bounds

14 An on-the-fly mark and sweep garbage collector based on sliding views
Hezi Azatchi, Yossi Levanoni, Harel Paz, Erez Petrank
October 2003 **ACM SIGPLAN Notices , Proceedings of the 18th annual ACM SIGPLAN conference on Object-oriented programing, systems, languages, and applications**, Volume 38 Issue 11
Full text available: pdf(244.12 KB) Additional Information: full citation, abstract, references, citings, index terms

With concurrent and garbage collected languages like Java and C# becoming popular, the need for a suitable non-intrusive, efficient, and concurrent multiprocessor garbage collector has become acute. We propose a novel mark and sweep on-the-fly algorithm based on the sliding views mechanism of Levanoni and Petrank. We have implemented our collector on the Jikes Java Virtual Machine running on a Netfinity multiprocessor and compared it to the concurrent algorithm and to the stop-the-world collecto ...

Keywords: concurrent garbage collection, garbage collection, memory management, on-the-fly garbage collection, runtime systems

15 Reducing garbage in Java
C. E. McDowell
September 1998 **ACM SIGPLAN Notices**, Volume 33 Issue 9
Full text available: pdf(289.15 KB) Additional Information: full citation, abstract, citings, index terms

One of the important advantages of Java, from a programmers prospective, is the use of garbage collection. One aspect of memory management in Java is that all objects are created on a garbage collected heap. Only primitive types, mostly numeric types and references to objects, are allocated on the runtime stack. We speculated that a significant number of objects behaved like traditional automatic variables, that are normally allocated on the runtime stack. We instrumented a Java virtual machine ...

16 Garbage collection and local variable type-precision and liveness in Java virtual machines
Ole Agesen, David Detlefs, J. Eliot Moss
May 1998 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1998 conference on Programming language design and implementation**, Volume 33 Issue 5
Full text available: pdf(1.54 MB) Additional Information: full citation, abstract, references, citings, index terms

Full precision in garbage collection implies retaining only those heap allocated objects that will actually be used in the future. Since full precision is not computable in general, garbage collectors use safe (i.e., conservative) approximations such as reachability from a set of root references. Ambiguous roots collectors (commonly called "conservative") can be overly conservative because they overestimate the root set, and thereby retain unexpectedly large amounts of garbage. We consider two m ...

17 MC^2: high-performance garbage collection for memory-constrained environments
Narendran Sachindran, J. Eliot B. Moss, Emery D. Berger
October 2004 **ACM SIGPLAN Notices , Proceedings of the 19th annual ACM SIGPLAN**

Conference on Object-oriented programming, systems, languages, and applications, Volume 39 Issue 10
Full text available: pdf(503.53 KB) Additional Information: full citation, abstract, references, index terms

Java is becoming an important platform for memory-constrained consumer devices such as PDAs and cellular phones, because it provides safety and portability. Since Java uses garbage collection, efficient garbage collectors that run in constrained memory are essential. Typical collection techniques used on these devices are mark-sweep and mark-compact. Mark-sweep collectors can provide good throughput and pause times but suffer from fragmentation. Mark-compact collectors prevent fragmentation, ...

Keywords: copying collector, generational collector, java, mark-compact, mark-copy, mark-sweep, memory-constrained copying

18 Pretenuring for Java
Stephen M. Blackburn, Sharad Singhai, Matthew Hertz, Kathryn S. McKinely, J. Eliot B. Moss
October 2001 **ACM SIGPLAN Notices , Proceedings of the 16th ACM SIGPLAN conference on Object oriented programming, systems, languages, and applications**, Volume 36 Issue 11
Full text available: pdf(389.44 KB) Additional Information: full citation, abstract, references, citings, index terms

Pretenuring can reduce copying costs in garbage collectors by allocating long-lived objects into regions that the garbage collector with rarely, if ever, collect. We extend previous work on pretenuring as follows. (1) We produce pretenuring advice that is neutral with respect to the garbage collector algorithm and configuration. We thus can and do combine advice from different applications. We find that predictions using object lifetimes at each allocation site in Java prgroams are accurate, whi ...

19 Using passive object garbage collection algorithms for garbage collection of active objects
Abhay Vardhan, Gul Agha
June 2002 **ACM SIGPLAN Notices , Proceedings of the 3rd international symposium on Memory management**, Volume 38 Issue 2 supplement
Full text available: pdf(192.46 KB) Additional Information: full citation, abstract, references, index terms

With the increasing use of active object systems, agents and concurrent object oriented languages like Java, the problem of garbage collection (GC) of unused resources has become more complex. Since active objects are autonomous computational agents, unlike passive object systems the criterion for identifying garbage in active objects cannot be based solely on reachability from a root set. This has led to development of specialized algorithms for GC of active objects. We reduce the problem of GC ...

Keywords: Java, active objects, actors, agents, garbage collection, program transformation

20 Implementing an on-the-fly garbage collector for Java
Tamar Domani, Elliot K. Kolodner, Ethan Lewis, Eliot E. Salant, Katherine Barabash, Itai Lahan, Yossi Levanoni, Erez Petrank, Igor Yanorer
October 2000 **ACM SIGPLAN Notices , Proceedings of the 2nd international symposium on Memory management**, Volume 36 Issue 1
Full text available: pdf(1.33 MB) Additional Information: full citation, abstract, citings, index terms

Java uses garbage collection (GC) for the automatic reclamation of computer memory no longer required by a running application. GC implementations for Java Virtual Machines

(JVM) are typically designed for single processor machines, and do not necessarily perform well for a server program with many threads running on a multiprocessor. We designed and implemented an on-the-fly GC, based on the algorithm of Doligez, Leroy and Gonthier [13, 12] (DLG), for Java in this environment. An *on-the-f ...*

***Keywords**: Java, concurrent garbage collection, garbage collection, memory management, on-the-fly garbage collection, programming languages*

Results 1 - 20 of 66 Result page: **1** 2 3 4 next

Home | Login | Logout | Access Information | Alerts |

Welcome United States Patent and Trademark Office

Search Results

BROWSE SEARCH IEEE XPLORE GUIDE

Results for "(garbage collection <in>ab) <and> (java<in>ab)"
Your search matched **34** of **1164322** documents.
A maximum of **100** results are displayed, **25** to a page, sorted by **Relevance** in **Descending** order.

e-mail

» View Session History

» New Search

» **Key**

IEEE JNL	IEEE Journal or Magazine
IEE JNL	IEE Journal or Magazine
IEEE CNF	IEEE Conference Proceeding
IEE CNF	IEE Conference Proceeding
IEEE STD	IEEE Standard

Modify Search

(garbage collection <in>ab) <and> (java<in>ab)

☐ Check to search only within this results set

Display Format: [✓ Citation] Citation & Abstract

Select Article Information

☐ 1. **An analysis of the garbage collection performance in Sun's HotSpot ™ Java Virt**
Dykstra, L.; Srisa-an, W.; Chang, J.M.;
Performance, Computing, and Communications Conference, 2002. 21st IEEE Internati
3-5 April 2002 Page(s):335 - 339
AbstractPlus | Full Text: PDF(563 KB) IEEE CNF

☐ 2. **The case for Java as a programming language**
Van Hoff, A.;
Internet Computing, IEEE
Volume 1, Issue 1, Jan.-Feb. 1997 Page(s):51 - 56
AbstractPlus | References | Full Text: PDF(200 KB) IEEE JNL

☐ 3. **An interactive environment for real-time software development**
Persson, P.; Hedin, G.;
Technology of Object-Oriented Languages, 2000. TOOLS 33. Proceedings. 33rd Interr
Conference on
5-8 June 2000 Page(s):57 - 68
AbstractPlus | Full Text: PDF(228 KB) IEEE CNF

☐ 4. **Real-time garbage collection for a multithreaded Java microcontroller**
Fuhrmann, S.; Pfeffer, M.; Kreuzinger, J.; Ungerer, T.; Brinkschulte, U.;
Object-Oriented Real-Time Distributed Computing, 2001. ISORC - 2001. Proceedings.
International Symposium on
2-4 May 2001 Page(s):69 - 76
AbstractPlus | Full Text: PDF(692 KB) IEEE CNF

☐ 5. **The object behavior of Java object-oriented database management systems**
Lo, C.-T.D.; Chang, M.; Frieder, O.; Grossman, D.;
Information Technology: Coding and Computing, 2002. Proceedings. International Con
8-10 April 2002 Page(s):247 - 252
AbstractPlus | Full Text: PDF(291 KB) IEEE CNF

☐ 6. **Scoped memory**
Bollella, G.; Reinholtz, K.;
Object-Oriented Real-Time Distributed Computing, 2002. (ISORC 2002). Proceedings.
International Symposium on
29 April-1 May 2002 Page(s):23 - 25

AbstractPlus | Full Text: PDF(224 KB) IEEE CNF

☐ 7. **Research and analysis of garbage collection mechanism for real-time embedded**
Liu Wei; Chen Zhang-long; Tu Shi-hang;
Computer Supported Cooperative Work in Design, 2004. Proceedings. The 8th Interna
on
Volume 1, 26-28 May 2004 Page(s):462 - 468 Vol.1
AbstractPlus | Full Text: PDF(675 KB) IEEE CNF

☐ 8. **Java and the shift to net-centric computing**
Hamilton, M.A.;
Computer
Volume 29, Issue 8, Aug. 1996 Page(s):31 - 39
AbstractPlus | References | Full Text: PDF(2860 KB) IEEE JNL

☐ 9. **A novel processor architecture with exact tag-free pointers**
Meyer, M.;
Micro, IEEE
Volume 24, Issue 3, May-June 2004 Page(s):46 - 55
AbstractPlus | Full Text: PDF(184 KB) IEEE JNL

☐ 10. **Who is collecting your Java garbage?**
Lo, C.-T.D.; Srisa-an, W.; Chang, J.M.;
IT Professional
Volume 5, Issue 2, March-April 2003 Page(s):44 - 50
AbstractPlus | Full Text: PDF(775 KB) IEEE JNL

☐ 11. **Supporting object accesses in a Java processor**
Vijaykrishnan, N.; Ranganathan, N.;
Computers and Digital Techniques, IEE Proceedings-
Volume 147, Issue 6, Nov. 2000 Page(s):435 - 443
AbstractPlus | Full Text: PDF(848 KB) IEE JNL

☐ 12. **NUMA-Aware Java Heaps for Server Applications**
Tikir, M.M.; Hollingsworth, J.K.;
Parallel and Distributed Processing Symposium, 2005. Proceedings. 19th IEEE Interna
04-08 April 2005 Page(s):108b - 108b
AbstractPlus | Full Text: PDF(504 KB) IEEE CNF

☐ 13. **The impact of realtime garbage collection on realtime Java programming**
Siebert, F.;
Object-Oriented Real-Time Distributed Computing, 2004. Proceedings. Seventh IEEE I
Symposium on
12-14 May 2004 Page(s):33 - 40
AbstractPlus | Full Text: PDF(1355 KB) IEEE CNF

☐ 14. **Adaptive QoS resource management in dynamic environments**
Chatterjee, S.; Brown, M.;
Multimedia Computing and Systems, 1999. IEEE International Conference on
Volume 2, 7-11 June 1999 Page(s):997 - 998 vol.2
AbstractPlus | Full Text: PDF(236 KB) IEEE CNF

☐ 15. **Hard real-time garbage collection in the Jamaica virtual machine**
Siebert, F.;
Real-Time Computing Systems and Applications, 1999. RTCSA '99. Sixth International
13-15 Dec. 1999 Page(s):96 - 102
AbstractPlus | Full Text: PDF(576 KB) IEEE CNF

16. **Theory versus practice in real-time computing with the Java^{TM} platform**
Foote, W.;
Object-Oriented Real-Time Distributed Computing, 1999. (ISORC '99) Proceedings. 2n
International Symposium on
2-5 May 1999 Page(s):105 - 108
AbstractPlus | Full Text: PDF(80 KB) IEEE CNF

17. **Consistency maintenance in Web-based real-time group editors**
Chengzheng Sun; Rok Sosic;
Electronic Commerce and Web-based Applications/Middleware, 1999. Proceedings. 1!
International Conference on Distributed Computing Systems Workshops on
31 May-4 June 1999 Page(s):15 - 22
AbstractPlus | Full Text: PDF(168 KB) IEEE CNF

18. **Dynamic memory management for real-time embedded Java chips**
Chi-Min Lin; Tien-Fu Chen;
Real-Time Computing Systems and Applications, 2000. Proceedings. Seventh Internat
on
12-14 Dec. 2000 Page(s):49 - 56
AbstractPlus | Full Text: PDF(656 KB) IEEE CNF

19. **Scalable hardware-algorithm for mark-sweep garbage collection**
Srisa-An, W.; Chia-Tien Dan Lo; Chang, J.M.;
Euromicro Conference, 2000. Proceedings of the 26th
Volume 1, 5-7 Sept. 2000 Page(s):274 - 281 vol.1
AbstractPlus | Full Text: PDF(648 KB) IEEE CNF

20. **A performance analysis of the active memory system**
Witawas Srisa-An; Srisa-an; Chia-Tien Dan Lo; J Morris Chang;
Computer Design, 2001. ICCD 2001. Proceedings. 2001 International Conference on
23-26 Sept. 2001 Page(s):493 - 496
AbstractPlus | Full Text: PDF(344 KB) IEEE CNF

21. **Cache performance in Java virtual machines: a study of constituent phases**
Rajan, A.S.; Shiwen Hu; Rubio, J.;
Workload Characterization, 2002. WWC-5. 2002 IEEE International Workshop on
25 Nov. 2002 Page(s):81 - 90
AbstractPlus | Full Text: PDF(683 KB) IEEE CNF

22. **A study on a garbage collector for embedded applications**
Krapf, R.C.; de Mattos, J.C.B.; Spellmeier, G.; Carro, L.;
Integrated Circuits and Systems Design, 2002. Proceedings. 15th Symposium on
9-14 Sept. 2002 Page(s):127 - 132
AbstractPlus | Full Text: PDF(251 KB) IEEE CNF

23. **A real-time Java system on a multithreaded Java microcontroller**
Pfeffer, M.; Uhrig, S.; Ungerer, T.; Brinkschulte, U.;
Object-Oriented Real-Time Distributed Computing, 2002. (ISORC 2002). Proceedings.
International Symposium on
29 April-1 May 2002 Page(s):34 - 41
AbstractPlus | Full Text: PDF(279 KB) IEEE CNF

24.
A multithreaded concurrent garbage collector parallelizing the new instruction in
Lo, C.-T.D.; Srisa-an, W.; Chang, J.M.;
Parallel and Distributed Processing Symposium., Proceedings International, IPDPS 20
CD-ROM

15-19 April 2002 Page(s):59 - 64
AbstractPlus | Full Text: PDF(213 KB) IEEE CNF

25. **A performance comparison between stop-the-world and multithreaded concurre garbage collection for Java**
Lo, C.-T.D.; Srisa-an, W.; Chang, J.M.;
Performance, Computing, and Communications Conference, 2002. 21st IEEE Internati
3-5 April 2002 Page(s):301 - 308
AbstractPlus | Full Text: PDF(748 KB) IEEE CNF